Dantzig-Wolfe Decomposition

A technique for handling large linear programs - breaks into a sequence of smaller LPs.

Consider the LP

\[
\begin{align*}
\text{min} & \quad c^T x \\
Ax &= b \\
Hx &= h \\
x &\geq 0
\end{align*}
\]

"hard constraint." A is m x n

"easy constraint." H is m x n.

Rewrite as

\[
\begin{align*}
\text{min} & \quad c^T x \\
Ax &= b \\
x \in X &\subseteq \text{"easy constraint"}
\end{align*}
\]

(\text{P})

Assume

\[X \subseteq \mathbb{R}^n_+\]

Solving LPs of the form \(\min c^T x \text{ } x \in X\)

is easy.

last solving the full LP \(\min c^T x \text{ } A x = b \quad x \in X\)

Now, \(X\) is a polyhedron.

\[S, \quad X = \{ x : x = \sum_{i=1}^p \lambda_i x_i + \sum_{j=1}^q \mu_j d_j, \lambda_i > 0, \mu_j > 0, \sum \lambda_i = 1 \} \]

where \(x_i\) are the extreme points of \(X\)

\(d_j\) are extreme rays of \(X\)
Def: \( d^2 \) is an extremum of \( X \) if it is a ray and
\[
d^2 = \lambda_1 d + \lambda_2 \hat{d}, \quad \lambda_1 > 0, \lambda_2 > 0, \quad d, \hat{d} \text{ rays}
\]
implies that \( \hat{d} = \mu_1 d^2, \hat{d} = \mu_2 d^2 \) for some \( \mu_1, \mu_2 > 0 \).

E.g.:

\[
\hat{d} \text{ is extremum ray. Since } X \in \mathbb{R}^2
\]

\[
\text{is not extremum ray. Scratch it!}
\]

\[
p \text{ is not extremum.}
\]

So, \((\ell)\) is equivalent to
\[
\min_{x, \lambda, \mu} c^T x
\]
so.
\[
Ax = b
\]
\[
x = \sum_{i} \lambda_i x_i + \sum_{j} \mu_j \hat{d}_j
\]
\[
\lambda \geq 0, \mu_j \geq 0, \Sigma \lambda = 1.
\]

I.e.,
\[
\min_{x, \lambda} c^T (\sum_{i} \lambda_i x_i + \sum_{j} \mu_j \hat{d}_j)
\]
so.
\[
A (\sum_{i} \lambda_i x_i + \sum_{j} \mu_j \hat{d}_j) = b
\]
\[
\sum_{i} \lambda_i \hat{d}_i = 1
\]
\[
\lambda \geq 0, \mu_j \geq 0
\]
The Master Problem (MP)

\[ \sum_{i=1}^{m} (c_i x_i) \lambda_i + \sum_{j=1}^{m} (c_j d_j) \mu_j \]

\[ \sum_{i=1}^{m} (A x_i) \lambda_i + \sum_{j=1}^{m} (A d_j) \mu_j = b \]

\[ \lambda_i \geq 0, \quad \mu_j \geq 0. \]

This is the **Master Problem**.

It has \( m+1 \) constraints, whereas \( 1 \) had \( m \) \( m \times 1 \) constraints.

But, in general, \( 1 \) considerably more than \( m \times 1 \).

**Example:**

\[ \text{min } -x_1 + x_2 + 4x_3 \]

\[ \text{st. } 2x_1 - 3x_2 + x_3 = 2 \]
\[ x_1 + x_2 - 3x_3 = 1 \]
\[ 0 \leq x_1 \leq 2, \quad 0 \leq x_2, \quad 0 \leq x_3 \leq 10 \]

\[ x, \lambda, \mu \]

Extremal points of \( x \): \( \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 2 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 2 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix} \]

Extremal \( x \): \( \begin{bmatrix} 0 \\ 0 \end{bmatrix} \)
\[ X = \sum_{x \in \mathbb{X}} x = \lambda_1 \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} + \lambda_2 \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \lambda_3 \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} + \lambda_4 \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} + \mu_1 \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \]

\[ \sum_{x \in \mathbb{X}} x = 1, \quad \lambda_1 \geq 0, \quad \mu_1 \geq 0 \]

**S. (MP) is**

\[ \begin{align*}
& \text{min} & & 0 \lambda_1 - 2 \lambda_2 + 40 \lambda_3 + 38 \lambda_4 + \mu_1 \\
& \text{s.t.} & & 0 \lambda_1 + 4 \lambda_2 + 10 \lambda_3 + 14 \lambda_4 - 3 \mu_1 \\
& & & 0 \lambda_1 + 2 \lambda_2 - 3 \lambda_3 - 2 \lambda_4 + \mu_1 \geq 2 \\
& & & \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 \\
& & & \lambda_1 \geq 0, \mu_1 \geq 0.
\end{align*} \]

\[ = \begin{bmatrix} 2 \\ 1 \end{bmatrix} \]

**Problem 2**

**Ded to (MP) is**

\[ \begin{align*}
& \text{max} & & b^T \pi + c \\
& \text{s.t.} & & (A_1 x) + c = c^T x, & i = 1, \ldots, p \quad (M_D) \\
& & & (A_2 x) + c = c^T d, & j = 1, \ldots, q
\end{align*} \]

**In example:**

\[ \begin{align*}
& \text{max} & & 2 \pi_1 + \pi_2 + c \\
& \text{s.t.} & & 4 \pi_1 + 2 \pi_2 + c \leq 0 \\
& & & 10 \pi_1 - 30 \pi_2 + c \leq -2 \\
& & & 14 \pi_1 - 28 \pi_2 + c \leq 40 \\
& & & -5 \pi_1 + \pi_2 \leq 1
\end{align*} \]
Have a large number of columns so don't want to calculate them or store them explicitly. So use revised simplex to solve (MP), only calculating columns as necessary.

At an iteration, have basic matrix $B$ for (MP), and $b_B, x_B$.

Calculate dual solution $\bar{y} = B^{-T}c_B$.

Then to check if $[\overline{y} \bar{x}]$ is optimal, need to look at reduced cost. This is equivalent to checking whether $[\overline{y} \bar{x}]$ is feasible in (MD). Now (MD) can be written:

$$\max \limits_{\pi, \xi} \ b^T \pi + \xi$$

s.t. $x^T (c - A^T \pi) \geq \xi \quad i = 1, \ldots, m \quad (M01)$

$$d_j^T (c - A^T \pi) \geq 0 \quad j = 1, \ldots, q.$$ 

This is equivalent to:

$$\max \limits_{\pi, \xi} \ b^T \pi + \xi$$

s.t. $x^T (c - A^T \pi) \geq \xi \quad \forall x \in X.$ 

(102)
Why?

Assume \((\tilde{\lambda}, \tilde{\delta})\) satisfies
\[
\begin{align*}
\tilde{\lambda}^T (c - A^T \tilde{\delta}) & \succeq \tilde{\delta} & i = 1, \ldots, p \\
\tilde{\delta}^T (c - A^T \tilde{\lambda}) & = 0 & j = 1, \ldots, q.
\end{align*}
\]

Then, for any \(\hat{x} \in X\),
\[
\hat{x} = \sum_i \lambda_i \hat{x}_i + \sum_j \tilde{\delta}_j \hat{d}_j
\]
so
\[
\hat{x}^T (c - A^T \tilde{\delta}) = \sum_i \lambda_i \hat{x}_i^T (c - A^T \tilde{\lambda}) + \sum_j \tilde{\delta}_j \hat{d}_j^T (c - A^T \tilde{\delta})
\]
\[
\geq \sum_i \lambda_i \hat{x}_i = \tilde{\delta}
\]

Conversely, assume \(\lambda, \delta\) satisfies
\[
\lambda^T (c - A^T \delta) \succeq \delta \quad \forall \lambda \in X
\]
Since \(\hat{x}\) is in \(X\), must have \(\hat{x}^T (c - A^T \delta) \succeq \delta\)

Also, if \(\hat{x} \in X\), then \(\hat{x} + \tilde{\delta}_j \hat{d}_j\) is \(X\) for any \(\tilde{\delta}_j \geq 0\), so
\[
(\hat{x} + \tilde{\delta}_j \hat{d}_j)^T (c - A^T \tilde{\lambda}) \succeq \tilde{\delta}_j \hat{d}_j \quad \forall \hat{d}_j \geq 0,
\]
so \(d_j^T (c - A^T \tilde{\lambda}) \succeq 0\).

Hence feasible region of \(\text{(MD1)}\) and \(\text{(MD2)}\) are identical.

Thus, checking whether \(\begin{bmatrix} \lambda \mid \delta \end{bmatrix}\) has reduced cost all nonnegative, and to check whether \(\begin{bmatrix} \lambda \mid \delta \end{bmatrix}\) satisfies
\[
(c - A^T \tilde{\delta})^T x \succeq \tilde{\delta} \quad \forall x \in X
\]

suggests solving the sub problem
\[
\begin{align*}
\min & \quad (c - A^T \tilde{\delta})^T x \\
\text{s.t.} & \quad x \in X.
\end{align*}
\]
There are three possible outcomes:

1) Optimal value of \((5P)\) is \(\geq \bar{z}\).
   Then \([\bar{x}_i] \) is feasible to \((MD)\), so \(\bar{x}_i \) must be optimal for \((MD)\).
   The optimal solution to \((5)\) is \(\bar{x}_i = \sum \bar{x}_i x^i + \sum \bar{x}_j d^j\).

2) Optimal value of \((5P)\) is finite but \(< \bar{z}\).
   Then there is an extreme point \(\bar{x}_i \) of \(X\), found by solving \((5P)\), with
   \(\sum (c - A^T \bar{x}_i) d_j \leq \bar{z}\).
   Reduced cost of the primal variable \(X\) corresponding to this extreme point is
   \[
   (c^T \bar{x}_i) - [w^T 1] [A^T \bar{x}_i]
   = c^T \bar{x}_i - w^T A \bar{x}_i - \bar{z} < 0.
   \]
   So this variable should enter the basis.

3) Optimal value of \((5P)\) is unbounded below.
   Then there is an extreme ray \(d_j \) of \(X\), found by solving \((5P)\), with
   \(\sum (c - A^T \bar{x}_i) d_j \leq \bar{z}\).
   Reduced cost of the primal variable \(X\) corresponding to this extreme point is
\[ c^T \bar{d}^j - \left[ \frac{1}{\pi^T} \bar{g} \right] \left[ \begin{array}{c} A^T \bar{d}^j \\ 0 \end{array} \right] = c^T \bar{d}^j - \frac{1}{\pi^T} A \bar{d}^j < 0. \]

So this variable should enter the basis.

Thus, all we need to keep around is the current basis. Don't need to have a list of all the extreme points and extreme rays of \( X \) — can generate them as needed.

See example.

Dantzig-Wolfe gets close to optimality quickly, but then takes a long time to finally converge.

\[ c^T X \]

\[ (c^T X)_i, \text{ for } X \in X. \]

\[ A X = b \]

For example, see over.
For our example, let us...
Remarks

1) What happens if we take $X = IR^n$?

One extreme point $x' = 0$

An extreme ray $d_k = cx_k$, let our vector $x_k$

So (LP):

$$\begin{align*}
\min & \quad (c^T x') \lambda_1 + \sum_{i=1}^n (A e_i) \mu_k \\
\text{s.t.} & \quad (A x') \lambda_1 + \sum_{i=1}^n (A e_i) \mu_k = b \\
\lambda_1 & = 1 \\
\lambda_1 \geq 0, \quad \mu_k \geq 0
\end{align*}$$

$$\begin{align*}
\min & \quad c^T x \\
\text{s.t.} & \quad A x = b \\
\lambda_1 & = 1 \\
\lambda_1 \geq 0, \quad \mu_k \geq 0
\end{align*}$$
2) \( X = [0,1]^n = \{ x : 0 \leq x \leq 1 \} \)
   The \( X \) has \( 2^n \) extreme points, 0 corners only.

3) Getting initial feasible solution:
   Use artificial variables.
   First get extreme point of \( X \) (eg., if it exists, solve \( \min c^T x \) ) Call it \( x_1 \)
   Phase I problem: Main Problem:
   \[ \begin{align*}
   \min & \quad c^T x \\
   \text{s.t.} & \quad \sum_{j=3} \lambda_j + \sum_{k=3} (A_k^y)^T \mu_k + (x_1^T \cdot 1) \omega = b \\
   & \quad \sum_j \lambda_j = 1 \\
   & \quad \lambda, \mu \geq 0, \omega > 0.
   \end{align*} \]
   Coeff of \( \omega \) is \( \sum 1 \) if \( b_i - (A x_1)_i > 0 \)
   \[ \frac{c}{\omega} \]
   So \( \omega = | b - (A x_1)_i | \) gives initial solution to Phase I problem.

Solve it, get basic feasible decomposition.
4) Prove block angular structure.

Consider LP of the form

\[ \min \quad c^T w^1 + c^T w^2 + \ldots + c^T w^p \]

subject to

\[ A_{01} w^1 + A_{02} w^2 + \ldots + A_{0p} w^p = b_0 \]

\[ A_{i1} w^1 + A_{i2} w^2 + \ldots + A_{ip} w^p = b_i, \quad i = 1, \ldots, m \]

\[ A_{p+1} w^p = b_p \]

\[ w^i \geq 0, \quad i = 1, \ldots, p \]

Let \( x^i = (w^1, w^2, \ldots, w^i) \), \( A_i x^i = b_i \), \( i = 1, \ldots, m \).

And \( u = (w^1, w^2, \ldots, w^p) \).

Extremal points of \( X \):

\[ x^i = \left( \begin{array}{c} x_{i1}^1 \\ x_{i2}^1 \\ \vdots \\ x_{ip} \end{array} \right) \]

Then \( x^i \) is an extremal point of \( X \).

Extremal rays of \( X \):

\[ d^i = \left( \begin{array}{c} 0 \\ 0 \\ \vdots \\ 0 \end{array} \right) \]

Extr ray of \( X \).
So solve \( SP_i(\tau) \) \( \min_{x \in X \tau} (c^T - \pi A_{0i})^T x \)
for all \( i \).

If no dual exists, given any \( X \) in most a basis.
If had optimal soln for all, then concatenation gives desired set \( \pi \) of \( X \).

Plug into (MP)

\[
\min \sum (c^T x_i) \lambda_j + \sum (d^T y_i) \mu_k
\]

\[
\sum (A_{0i} x_i) \lambda_j + \sum (A d^T) \mu_k = b
\]

\[
\sum \lambda_j \mu_k = 1
\]

\[
\lambda_j \geq 0, \mu_k \geq 0.
\]

Use \( c = (c_1, \ldots, c_p) \)

\[ A_0 = (A_{01}, \ldots, A_{0p}) \]

Alternative: Substitute for each \( u \in X \), not for \( u \in X \)

The (MP) is:

\[
\min \sum_{j \in J, i} (c^T x_i) \lambda_{j} + \sum_{k \in K, i} (d^T y_i) \mu_{k} + \sum_{j \in J, i} (A_{0i} x_i) \lambda_{j} + \sum_{k \in K, i} (A d^T) \mu_{k} = b
\]

\[
\sum_{j \in J, i} \lambda_{j} = 1
\]

\[
\sum_{k \in K, i} \mu_{k} = 1
\]

\[
\lambda_{j} \geq 0, \mu_{k} \geq 0.
\]
Advantages:

1) May only have to solve few subproblems to get new column for master problem.
2) Each extreme point of each $X_i$ can have weight many independently of other sets of being tied with others.

5) Other block angular structures:

\[ A = \begin{bmatrix} I & 0 \\ 0 & J \end{bmatrix} \]

Diagonal block angular

Apply duality to dual.

\[ A = \begin{bmatrix} I & P \\ 0 & J \end{bmatrix} \]

Subproblem: $A_1 = \begin{bmatrix} I & P \\ 0 & J \end{bmatrix}$

Solve subproblem by decomposition applied to dual.

Do much more certain problems: Nested decomposition.
Eg: \ldots
Each sub-problem must be solved with single period constraints and proposals generated by future periods.

This is a typical production & inventory model:

\[ P_i = \text{production in period } i \]
\[ S_i = \text{storage in period } i \]
\[ I_i = \text{inventory available in period } i \]
\[ b_i = \text{requirement in period } i \]
6) Don't have to solve subproblem to optimality:
   \[ (c-Ax)\ x < 0 \]
   then \( x^* = x \) OK.

7) Bender Decomposition: generate constraints by looking at
   extreme rays of dual polyhedron.

8) Oracle: Don't have black electronics. Have oracle solve subroutines.

Combinatorial experience with decomposition.


Comparisons with simplex.

Disappears in general, but better for really big problems.

If subproblem "nice", e.g. network constraints, then can be
   competitive with revised simplex on (1).

Acts close to optimal quickly, but takes a long time to
   finally converge.
We can't have the subproblem value $\bar{v}$.

$S^1(\pi) = \max \quad (c-A\pi^*)^T x$

subject to $x \in X$.

$D^n(\pi)$:

$\max \quad \sum_i (c^T x^i) \lambda_i + \sum_j (d^T d^j) \mu_j$

subject to

$\sum_i (A x^i)^T \lambda_i + \sum_j (d d^j)^T \mu_j = b$

$\sum \lambda_i = 1$

$\lambda \geq 0$, $\mu \geq 0$

(RMD)

$\max \quad b^T \pi + \bar{v}$

subject to

$(A x)^T \pi + \bar{v} \leq c^T x^i$

$(A x)^T \pi \leq c^T d$

Any optimal basis for (RMP) must include at least one $x^i$.

The corresponding constraints in (RMD) hold, at equality, by construction of $\pi, \bar{v}$

so

$(A x^i)^T \pi + \bar{v} = c^T x^i$ for this $x^i$.

Therefore

$\bar{v} = (c-A\pi^*)^T x^i$ for this $x^i$.

Since $x^i \in X$, the optimal value $v$ for $S^1(\pi)$ is

$v \leq (c-A\pi^*)^T x^i = \bar{v}$. 

Relationship with Lagrangian Relaxation:

\[
\begin{align*}
\text{Relax} & \quad \min c^T x \quad \text{to} \quad z(\text{min} \\ \text{s.t.} \quad Ax = b \\ \quad U x = 0 \\ \quad x \geq 0)
\end{align*}
\]

Lagrangian dual is \[ \max z(\lambda) \]

For each \( j \), dual is \[ \Lambda_j(\lambda) \]

\[
\begin{align*}
\max & \; g_j^T x + b_j^T \lambda \\
\text{s.t.} & \; W_j \leq c - A^T x \\
& \; x \geq 0 \\
& \; \lambda \geq 0
\end{align*}
\]

Now, could write dual to be:

\[
\begin{align*}
\max & \; z + b^T \lambda \\
\text{s.t.} & \; z \leq c^T x + A^T (\gamma - A x) \quad \text{for all } x \text{ satisfying } U x = 0, x \geq 0
\end{align*}
\]

or

\[
\begin{align*}
\max & \; c^T x - (A^T \lambda)^T x \\
\text{s.t.} & \; U x = 0 \\
& \; x \geq 0
\end{align*}
\]

Give constraints of the form:

\[
\begin{align*}
z & \leq c^T x + (A^T \lambda)^T x \\
0 & \leq c^T x - (A^T \lambda)^T x
\end{align*}
\]

for all extreme points.
Logarithmic relaxation for convex NLPs

\[ \min \frac{f(x)}{g_i(x)} \leq 0 \quad i = 1, \ldots, m \quad \text{(NLP)} \]

\[ \min f(x) + \lambda^T g(x) \quad \text{(NLP(x))} \]

\[ \max z \quad \text{subject to} \quad f(x) + \lambda^T g(x) \leq z \quad \forall x \in X \]

Get subproblem of form \((NLP(x))\).

Each new \(x^i\) or \(d^j\) gives a constraint on \((\Omega)\):

\[ \max z \quad \text{subject to} \quad \begin{align*}
    f(x^i) + \lambda^T g(x^i) &\leq z \\
    \lambda &\geq 0
\end{align*} \quad \text{for } x^i \text{ generated} \]

If subproblem is unbounded, need to construct constraint just in \(d\).
Cutting Stock Problem. (Chapter 13)

Minimise the number of cuts needed for each product.

E.g., paper, textiles, foil, cellophane, silk, ... E.g.: r = 50, w = 14, 18, 21.

The formulation (BAD)

Let $x_{ij} =$ # units of width $w_i$ from raw roll $j$.

Let $z_j = \begin{cases} 1 & \text{if used roll } j \\ 0 & \text{otherwise} \end{cases}$

Let $N$ be upper bound on number of rolls we need.

min $\sum_{j=1}^{N} z_j$

s.t. $\sum_{i} w_i x_{ij} \leq a_i z_j$ for $j=1, ..., N$

s.t. $\sum_{j} x_{ij} \geq b_i$ for demands $i = 1, ..., m$

$\text{BAD becomes:}$

(i) Symmetry: As soon as we fix $x_{ij}$ for one roll, that fractional solution can now be a different roll.

(ii) Need to go back to the true $Lx = z$; too easy to get $Lx = z$, because we throw away something for most rolls.
Column generation formulation (Good).

Let $a^j$ denote a cutting pattern, $j = 1, \ldots, p$.

E.g.: $c = 50$, $w_i = 14, 18, 21$

Possible patterns are:

- $(3, 0, 0)$: waste $8$
- $(0, 2, 0)$: waste $14$
- $(0, 0, 2)$: waste $8$
- $(1, 2, 0)$: waste $0$
- $(2, 0, 1)$: waste $1$
- $(0, 1, 1)$: waste $11$

Let $\mathbf{x}^j$ = number cut into pattern $j$.

Solve: $\min \sum_{j=1}^{p} x^j$ \hspace{1cm} (CSP) \hspace{1cm} May be good to write down this formulation for the six patterns above.

subject to $\sum_{j=1}^{p} a^j x^j = b$

$x^j \geq 0$, integer.

Could have a very large number of possible patterns, so

use column generation.

Ignore integrality restriction, and try to round LP solution at end to get good solution.

(Note: rounding may not give optimal integral solution.)
Let, with $r$ and $u < 2$ above, say $b = (10, 15, 8)$.

So, we have with patterns $a_1 = (3, 0, 0), a_2 = (0, 2, 0), a_3 = (0, 0, 2)$,

\[
x = \left( \frac{10}{3}, \frac{15}{2}, \frac{8}{3} \right)
\]

\[
\begin{align*}
\sum x_i + x_3 &= 10 \\
3x_1 &= 10 \\
2x_2 &= 15 \\
2x_3 &= 8 \\
x_i &\geq 0.
\end{align*}
\]

Due to (CSP),

\[
\begin{align*}
\max & \quad \sum_{i=1}^{m} b_i y_i \\
\text{s.t.} & \quad a_i^T y_i \leq 1
\end{align*}
\]

(\text{CSD})

Thus to determine if our dual feasible, we need to check whether $a_i^T y_i \leq 1$ for all possible patterns $a_i$.

In example, $B = \begin{bmatrix} 3 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix}$ and $c_8 = \begin{bmatrix} 11 \\ 1 \\ 1 \end{bmatrix}$,

\[
y = B^{-T} c_8 = \begin{bmatrix} \frac{1}{3} \\ \frac{1}{2} \\ \frac{1}{2} \end{bmatrix}
\]

So, check whether $\frac{1}{3} a_1 + \frac{1}{2} a_2 + \frac{1}{2} a_3 \leq 1$ for all patterns $a_i$. 

Suggest solving:

\[
\max \quad \frac{1}{3} a_1 + \frac{1}{2} a_2 + \frac{1}{4} a_3 \\
\text{s.t.} \quad (a_1, a_2, a_3) \text{ is a valid cutting pattern}
\]
or:

\[
\max \quad \frac{1}{3} a_1 + \frac{1}{2} a_2 + \frac{1}{4} a_3 \\
\text{s.t.} \quad 14 a_1 + 18 a_2 + 21 a_3 \leq 50 \\
\quad \quad \quad a_i \text{ integer, } \geq 0.
\]

This is a Knapsack problem: NP-complete, so use heuristics. See Chapter 9 for example algorithms.

\[ a^* = (2, 0, 1) \] is feasible linear problem, with value \( \frac{2}{3} + \frac{1}{2} = \frac{7}{6} \geq 1 \)

So this gives an extra column:

\[
\min \quad x_1 + x_2 + x_3 + x_4 \\
\text{s.t.} \quad 3x_1 + 2x_4 = 10 \\
\quad \quad 2x_2 = 15 \\
\quad \quad 2x_3 + x_4 = 8 \\
\quad \quad x_i \geq 0.
\]
Need to find $B^{-1}a^T$ in order to determine leaving variable.

$$B^{-1}a^T = \begin{bmatrix} -\frac{3}{2} \\ 0 \\ \frac{1}{2} \end{bmatrix} \\
B^{-1}b = \begin{bmatrix} -\frac{10}{3} \\ 15 \frac{1}{2} \\ 4 \end{bmatrix} \quad \text{ratio:} \quad - \frac{10}{3}$$

So $x_1$ leaves basis, get $B = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \end{bmatrix}, c_B = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

Solve $Bx = b$:

$$x = \begin{bmatrix} \frac{5}{2} \\ \frac{15}{2} \\ \frac{1}{2} \end{bmatrix}$$

Primal values: $x_1 = \frac{5}{2}, x_2 = \frac{15}{2}, x_3 = \frac{1}{2}$.

Solve $B^T y = c_B$:

$$y = \begin{bmatrix} \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \Rightarrow y_2 = \frac{1}{2}, y_3 = 0, y_4 = \frac{1}{2}.$$

To check dual feasibility, solve:

$$\max \quad \frac{1}{4} a_1 + \frac{1}{2} a_2 + \frac{1}{4} a_3$$

$s.t. \quad 14 a_1 + 18 a_2 + 21 a_3 \leq 50$

$a_i \geq 0, a_i \in \mathbb{R}.$

One feasible solution: $a = (1, 2, 0)$, value $\frac{1}{4} > 1$.

5. add this column to problem:

$$\min \quad x_1 + x_2 + x_3 + x_4 + x_5$$

$s.t. \quad 3x_1 + 2x_4 + x_5 = 10$

$$2x_6 + 2x_5 = 15$$

$$2x_3 + x_4 = 8$$

$x_i \geq 0$. 
Need to find $B^{-1} a^x = a^x$:

\[
\begin{bmatrix}
2 & 0 & 0 \\
0 & 2 & 0 \\
1 & 0 & 2 \\
\end{bmatrix} \times a^x = \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix} \Rightarrow a^x = \begin{bmatrix} \frac{1}{2} \\ 1 \\ \frac{1}{2} \end{bmatrix}
\]

So, middle row, $x_3 \times 2$, leave basis:

\[
B = \begin{bmatrix}
2 & 0 & 0 \\
0 & 2 & 0 \\
0 & 0 & 2 \\
\end{bmatrix}
\]

Solve $B x = b$:

\[
x = \begin{bmatrix}
x_1 \\ x_2 \\ x_3 \\
\end{bmatrix} = \begin{bmatrix}
\frac{5}{4} \\ \frac{15}{2} \\ 27/8 \\
\end{bmatrix}
\]

Value: $\frac{5}{4} + \frac{15}{2} + \frac{27}{8} = \frac{97}{8} = 12.125$

Solve $B^T y = c^x$:

\[
\begin{bmatrix}
2 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 2 \\
\end{bmatrix} \times y = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \Rightarrow y = \begin{bmatrix} \frac{4}{3} \\ \frac{3}{8} \\ \frac{1}{2} \end{bmatrix}
\]

To check dual feasibility, solve:

\[
\max \; \frac{1}{4} a_1 + \frac{3}{8} a_2 + \frac{1}{2} a_3
\]

s.t.

\[
14 a_1 + 18 a_2 + 21 a_3 \leq 50
\]

\[a_i \; \text{integer}, \; \geq 0.\]

All the patterns listed earlier have value $\leq 1$. So, no appear to be dual feasible, so we've solved the LP relaxation.
Getting an integer solution:

Our LP solution is: 5/4 of pattern (2, 0, 1)
1/4 of pattern (1, 2, 0)
2/4 of pattern (0, 0, 2)

Uses 12 3/4 rows.

Could round up: 2 of pattern (2, 0, 1)
8 of pattern (1, 2, 0)
4 of pattern (0, 0, 2)

Uses 14 rows.

Produces (12, 16, 10).

(wanted 10, 15, 8.)

Could round down: 1 of pattern (2, 0, 1)
7 of pattern (1, 2, 0)
3 of pattern (0, 0, 2).

Uses 11 rows.

Produces (9, 14, 7)

Need additional (1, 1, 1).

So use one each of patterns
(1, 1, 0)
(0, 0, 1)

Use 13 rows altogether.

Optimal since LP relaxation has value 12 3/4, so integer solution must need at least \( \lceil 12 \frac{3}{4} \rceil = 13 \) rows.

Note: rounded-up stalls used or more in rows more than the optimal, you lose a positive variance.
Crew Scheduling

Clique Partitioning
Network Simplex Algorithm

The max flow problem is an example of an upper-bounded transshipment problem.

\[
\begin{align*}
& \text{min } c^T x \\
& \text{subject to } Ax = b \quad A \text{ an } m \times n \text{ matrix, } A_{\text{inc}} \text{ incidence matrix of a graph, } b = \text{ vector of demands}. \\
& 0 \leq x \leq a
\end{align*}
\]

when \( A \) is incidence matrix of a graph, \( b = \) vector of demands.

Each column of \( A \) has one \( +1 \), one \( -1 \), so \( \sum c_{ij} = 0 \) \( \forall j \).

Total supply = total demand, so \( \sum b_i = 0 \).

Thus, rank \((A, b)\) = \( m - 1 \), when \( m = \# \) vertices.

So we can delete one row.

Then a basic matrix will have \( m - 1 \) columns.

Thus the edge of the columns in a basic matrix form a bfs.

Thus the edges constituting the basic variables to a bfs must constitute a tree.

\[\text{Assume not. So must have a cycle. Consider these columns of } A:\]

\[
\begin{bmatrix}
1 & 1 & 1 & -1 & -1 & -1 \\
1 & -1 & -1 & 1 & 1 & 1 \\
1 & 1 & -1 & -1 & 1 & 1 \\
1 & 1 & 1 & -1 & -1 & -1 \\
1 & 1 & 1 & 1 & -1 & -1
\end{bmatrix}
\]

Cols are linearly dependent.
Find reduced costs:

Dual problem: \( \max b^T \pi - u^T y \)
\[ \text{s.t. } A^T \pi - y \leq c \]
\[ y \geq 0. \]

If \( x_e \) basic then \( y_e = 0 \), and \( (A^T \pi)_e = c_e \), for complementary slackness.

So solve \( B^T \pi = c \) for basic edges

Easy, because only two nonzero for each column of \( B \), and each column only has one nonzero.

Are we dual feasible?

Allow \( y_e > 0 \) only if \( x_e \) at its upper bound.

So for \( x_e = 0 \), nonbasic, check \( (B^T \pi)_e \leq c_e \)
\( x_e = u_e \), nonbasic, check \( (B^T \pi)_e \geq c_e \), since
\[ (B^T \pi)_e - y_e = c_e \] for these variables.

If have negative reduced cost, introduce arc:
Create cycle:

In picture, assume arc are at their lower bound.

Min ratio test: Find smallest \( \epsilon \) so that keep \( 0 \leq x_e \leq u_e \).
Throw that are out. Still have a spanning tree. Repeat.
Not all augmenting paths can be interpreted as simplex steps. However, the one we did before can be:

Only two edges must be basic; otherwise, all have $x_{ij} = 0$ or $c_{ij}$. So, arbitrarily designate remaining basic $x_{ij}$.

(Not every arbitrary designation will immediately lead to improvement, because of degeneracy.)

Do pay, NSA 4 work.

Choose $x_{ij}$ so that \[
\frac{c_{ij}}{a_{ij}} = c_{ij} \text{ for basic edges.}
\]

Reduced cost: \[
\frac{c_{ij} + \bar{a}_{ij}}{a_{ij}} \text{ in } a_{ij}.
\]

Only reduced cost is as indicated, so decrease flow on that arc, giving the update we got before.
Here, the primal problem is:

\[
\begin{align*}
\text{min} & \quad \sum c_{ij} x_{ij} \\
\text{s.t.} & \quad \sum_j x_{ij} + \sum_j y_{ij} = 0 \\
& \quad 0 \leq x_{ij} \leq u_{ij} \\
\end{align*}
\]

Dual: \[
\begin{align*}
\text{max} & \quad -\sum_{ij} c_{ij} y_{ij} \\
\text{s.t.} & \quad -\pi_i + \pi_j - y_{ij} \leq c_{ij} \\
& \quad y_{ij} \geq 0,
\end{align*}
\]

Complementary slackness:

\[
\begin{align*}
x_{ij} \left( c_{ij} + \pi_i - \pi_j + y_{ij} \right) &= 0 \\
y_{ij} \left( u_{ij} - x_{ij} \right) &= 0.
\end{align*}
\]

Three possibilities for \(x_{ij}\):

1. \(x_{ij} \text{ basic} \Rightarrow y_{ij} = 0, \quad -\pi_i + \pi_j = c_{ij}, \quad i.e. \quad c_{ij} + \pi_i - \pi_j = 0\)
2. \(x_{ij} = 0, \text{ nonbasic} \Rightarrow y_{ij} = 0, \quad -\pi_i + \pi_j \leq c_{ij}, \quad i.e. \quad c_{ij} + \pi_i - \pi_j \geq 0\)
3. \(x_{ij} = u_{ij}, \text{ nonbasic} \Rightarrow \\
-\pi_i + \pi_j - y_{ij} = c_{ij}, \\
\therefore \quad -\pi_i + \pi_j \geq c_{ij}, \quad i.e. \quad c_{ij} + \pi_i - \pi_j \leq 0\).

Thus, for optimality, we need

\[
\begin{align*}
c_{ij} + \pi_i - \pi_j \geq 0 & \text{ if } x_{ij} = 0 \\
c_{ij} + \pi_i - \pi_j \leq 0 & \text{ if } x_{ij} = u_{ij} \\
c_{ij} + \pi_i - \pi_j = 0 & \text{ if } x_{ij} \text{ basic.}
\end{align*}
\]

Choose \(y_{ij}\) to satisfy this.

Then check the other two.
Multicommodity Network Flow Problem

(Network Flows, Ahuja, Magnanti, Orlin; Chapter 17).

See handout with applications.

Several commodities flow through the network; each commodity has its own set of origins & destinations.

Each commodity has limited total capacity $u_{ij}$.

Let $N = \text{node arc incidence matrix}$ (give small example).

Then we can express the multicommodity network flow problem as:

\[
\min \sum_{1 \leq k \leq K} \sum_{(i,j) \in A} c_{ij} x_{ij}^k
\]

s.t. \[
\sum_{1 \leq k \leq K} x_{ij}^k \leq u_{ij} \quad \forall (i,j) \in A
\]

\[
W x^k = b^k \quad \text{for} \ 1 \leq k \leq K
\]

\[
0 \leq x_{ij}^k \leq u_{ij}^k \quad \text{when the} \ k\text{-th commodity, on}
\]

arc $i \rightarrow j$.
and \( x_{ij} \) = flow of commodity \( l \) on arc \((i, j)\)

\( c_{ij} \) = cost of transporting one unit of commodity \( l \) on arc \((i, j)\)

\( A \) = set of arcs,

\( b_k \) = origin/destination requirements of commodity \( l \).

Note: may well have fractional optimal solutions.

**Column generation approach.** (§17.5, AMO)

Assume all arc costs are nonnegative, so no cycles in optimal solution.
We can represent any flow as a sum of paths.

E.g.
So reformulate as,

\[ m_k \sum_{1 \leq k \leq K} \sum_{P \in \mathcal{P}_k} c_k(P) f(P) \]  \tag{1}

s.t.

\[ \sum_{1 \leq k \leq K} \sum_{P \in \mathcal{P}_k} \delta_{ij}(P) f(P) = u_{ij} \quad \forall (i, j) \in A \] \tag{2}

\[ \sum_{P \in \mathcal{P}_k} f(P) = d_k \] \tag{3}

\[ f(P) \geq 0 \]

when \( f(P) \) is the flow of commodity \( k \) on path \( P \),

\[ \delta_{ij}(P) = \begin{cases} 1 & (i, j) \in P \\ 0 & \text{otherwise} \end{cases} \]

Has \( m + K \) constraints.

The arc formulation has \( m \times n \) constraints.

E.g. \( m = 1000, n = 5000, \lambda = 10^{-6} \) for each pair of nodes.
Note: Matrix let structure:

\[
\begin{bmatrix}
\text{m rows} \\
\text{K rows}
\end{bmatrix}
\]

Now, \# of basic variables = m + K.

Need at least one basic variable for each of the K constraints.

So only need to maintain m basic variables - other basic variables can be found from equation (3).

So: can use the Generalized Upper Bounding Simplex Method.

Even thought can perform all the matrix computation on a matrix
of size mKm, not (m+K)(m+K)
(ie 5000x5000, not 100K000x100K000 for example)

Doubtful: exponential number of variables.

At most K times paths very positive thus in an optimal solution.

Solution uses 2 or more paths for at most m constraints, jitter on path for remaining constraints.
Generating plans (i.e., paths)

Dual formulation:

\[
\begin{align*}
\max & \quad -\sum_{(i,j) \in A} w_{ij} u_{ij} + \sum_{k=1}^{K} \delta_k d_k \\
\text{s.t.} & \quad -\sum_{(i,j) \in P} w_{ij} u_{ij} + \delta_k \leq c^{u}(P) \quad \forall \text{path } P \in P_k, \\
& \quad \delta_k \geq 0,
\end{align*}
\]

Reduced cost:

\[c^{u}(P) + \sum_{i,j} w_{ij} - \delta_k,
\]

where \(w_{ij}, \delta_k\) are current dual solutions.

Complementary slackness:

\[
\begin{align*}
\delta_k \left( u_{ij} - \sum_{i,j} \delta_k c^{u}_i (P) f(i) \right) &= 0 \quad \text{Price} = 0 \quad \text{if} \quad \delta_k > 0 \\
\left( c^{u}(P) + \sum_{i,j} w_{ij} - \delta_k \right) f(P) &= 0.
\end{align*}
\]

This determines \(\delta_k\), if any, simplex.

So \(\delta_k\) is the shortest path distance from \(s_k\) to \(t_k\) with respect to the modified cost \(c^{u}_i + w_{ij}\), and in the optimal solution every path from \(s_k\) to \(t_k\) that carries a positive flow must be a shortest path with respect to the modified costs.

If \(\delta_k\) is not shortest path (i.e., \(3P \
eq \min \sum_{i,j} (c^{u}_i + w_{ij}) < \delta_k\), i.e., with negative reduced cost
So to find new column with negative cost.

Look for shortest path from $s_k$ to $b_k$ with the modified weight $e_{ij} + w_{ij}$.

If shortest path is $\leq 6$ then have $-w$ red cost.

Lower bound

Lagrangian dual:

$$\max \quad \theta(\bar{\lambda}) = \inf L(\bar{t}, \bar{\lambda}, \bar{\mu})$$

s.t. $\Sigma \mu_i = d \bar{\lambda}$

$$\bar{t}(i) = 0.$$  

$$= \sum_{i,j \in A} \nu_{ij} - \sum_{i,j \in A} e_{ij} \mu_{ij}$$

where $\nu_{ij} = \text{length of shortest path with cost } e_{ij} + w_{ij}$.  

MCNF C